

Optimal Geo-Indistinguishable Mechanisms for Location Privacy

Nicolás E. Bordenabe^{1,3}, Konstantinos Chatzikokolakis^{2,3},
and Catuscia Palamidessi^{1,3}

¹ INRIA

² CNRS

³ LIX, Ecole Polytechnique

Abstract. With location-based services becoming increasingly more popular, serious concerns are being raised about the potential privacy breaches that the disclosure of location information may induce. We consider two approaches that have been proposed to limit and control the privacy loss: one is the geo-indistinguishability notion of Andrés et al., which is inspired by differential privacy, and like the latter it is independent from the side knowledge of the adversary, and robust with respect to composition of attacks. The other one is the mechanism of Shokri et al., which offers an optimal trade-off between the loss of quality of service and the privacy protection with respect to a given Bayesian adversary. We show that it is possible to combine the advantages of the two approaches: given a minimum threshold for the degree of geo-indistinguishability, we construct a mechanism that offers the maximal utility, as the solution of a linear program. Thanks to the fact that geo-indistinguishability is insensitive to the remapping of a Bayesian adversary, the mechanism so constructed is optimal also in the sense of Shokri et al. Furthermore we propose a method to reduce the number of constraints of the linear program from cubic to quadratic (with respect to the number of locations), maintaining the privacy guarantees without affecting significantly the utility of the generated mechanism. This lowers considerably the time required to solve the linear program, thus enlarging significantly the size of location sets for which the optimal trade-off mechanisms can still be computed.

1 Introduction

While location-based systems (LBSs) have demonstrated to provide enormous benefits to individuals and society, these benefits come at the cost of users' privacy: as discussed in [1,2,3], location data can be easily linked to a variety of other information about an individual, and expose sensitive aspects of her private life such as her home address, her political views, her religious practices, etc.. There is, therefore, a growing interest in the development of location-privacy protection mechanisms (LPPMs), that allow to use SBSs while providing sufficient privacy guarantees for the user. Most of the approaches in the literature are based on

perturbing somehow the information reported to the LBS provider, in order to make it difficult for the adversary to infer the user’s true location [4,5,6,7,8,9].

Clearly, the perturbation of the information sent to the LBS provider leads to a degradation of the quality of service, and consequently there is a trade-off between the level of privacy that the user wishes to guarantee and the service quality loss (SQL) that she will have to accept. The study of this trade-off, and the design of mechanisms which optimize it, is an important research direction started with the seminal paper of Shroki et al. [8].

Obviously, any such study must be based on meaningful notions of privacy and SQL. The authors of [8] consider the privacy threats deriving from a Bayesian adversary. More specifically, they assume that the adversary knows the prior probability distribution on the user’s possible locations, and they quantify privacy as the expected distance between the true location and the best guess of the adversary once she knows the location reported to the LBS. This guess takes into account the information already in her possession (the prior probability), and it is by definition more accurate, in average, than the reported location. We also say that the adversary may *remap* the reported location.

The notion of quality loss adopted in [8] is also defined in terms of the expected distance between the real location and the reported location, with the important difference that the LBS is not assumed to know the user’s prior distribution (in fact, in general it is not tuned for any specific user), and consequently it does not apply any remapping. For this reason, when the notion of distance is the same, the SQL is always greater than or equal to the privacy. The optimal mechanism in [8] is defined as the one which maximizes privacy for a given SQL threshold, and since these measures are linear functions of the noise (characterized by the conditional probabilities of each reported location given a true location), such mechanism can be computed by solving a linear optimization problem.

In this paper, we also consider the geo-indistinguishability framework of [9], another notion of privacy which is based on differential privacy [10], and more precisely, on the extension of differential privacy to arbitrary metrics proposed in [11]. Intuitively, a mechanism provides geo-indistinguishability if two locations that are geographically close have similar probabilities to generate a certain reported location. Equivalently, the reported location will not increase by much the adversary’s chance to distinguish the true location among the nearby ones. Note that this notion protects the accuracy of the location: the adversary is allowed to distinguish locations which are far away. It is important to note that the property of geo-indistinguishability does not depend on the prior. This is a feature inherited from differential privacy, which makes the mechanism robust with respect to composition of attacks in the same sense of differential privacy [11].⁴

⁴ This does not mean that the prior knowledge is not harmful for privacy: it is harmful, for both geo-indistinguishability and differential privacy, as it is for any obfuscation mechanism. But the compositionality guarantees that if the prior knowledge is ac-

In this paper we aim at combining the advantages of the two above approaches to privacy protection. Namely, we aim at enhancing the optimal mechanism of [8], whose optimality and privacy level hold for a specific prior only, with privacy guarantees that are independent from the prior. Consider, for instance, a user for which the optimal mechanism has been computed with respect to his average day (and consequent prior p), and who has very different habits in the morning and in the afternoon. By simply taking into account the time of the day, the adversary can use a different prior, and make a much more effective attack, and consequently the privacy guarantees that the mechanism provides for p can be violated in an uncontrolled way when the adversary has some additional knowledge. Choosing a mechanism that, besides being optimal for the given prior, provides also geo-indistinguishability, would add an additional shield against the privacy loss.

In order to achieve this goal, we fix a lower bound on the level of geo-indistinguishability, and we compute a mechanism K with minimum SQL among those which respect the bound. The property of geo-indistinguishability is expressed by linear constraints, hence we can generate K by solving a linear optimization problem (note that this linear optimization problem is different from the one in [8]). Since geo-indistinguishability is not affected by remapping, the expected error of the adversary must coincide with the SQL, i.e., the adversary cannot gain anything by any remapping H , or otherwise KH would be still geo-indistinguishable and provide a better SQL. Since the privacy coincide with the SQL, it must be maximum. In conclusion, we obtain a geo-indistinguishable K with minimum SQL and maximum privacy (for the given SQL).

Note that the optimal mechanisms are not unique, and ours does not usually coincide with the one produced by the algorithm of [8]. In particular the one of [8] in general does not provide geo-indistinguishability, while ours does, by design. The robustness of the geo-indistinguishability property seems to affect favorably also other notions of privacy: We have evaluated the two mechanisms with the privacy definition of [8] on some specific real data, and we have observed that, while they obviously coincide on the prior for which the mechanism of [8] is computed, ours performs significantly better when we consider different priors.

It is worth noting that the amount of linear constraints required to express geo-indistinguishability is, in general, cubic with respect to the number of locations considered. We present an approximation technique to reduce considerably this number (from cubic to quadratic), based on the use of a metric induced by a spanning graph of the set of locations. This way, instead of considering the geo-indistinguishability constraints for every pair of locations, we only consider those for every edge in the spanning graph. We also show, based on experimental results, that for a reasonably good approximation our approach offers an improvement in running time with respect to method of Shokri et al. We must note however that the mechanism obtained this way is no longer optimal with respect to the original metric, but to the metric induced by the graph instead, and

quired only via differentially private mechanisms, then the loss of privacy is gradual and under control.

therefore the SQL of the mechanism might be higher, although our experiments also show that this increase is not significant.

Contribution The main contributions of this paper are the following:

- We present a method based on linear optimization to generate a mechanism that is geo-indistinguishable and achieves optimal utility. Furthermore the mechanism is also optimal with respect to the expected error of the adversary.
- We evaluate our approach under different priors (generated from real traces), and show that it outperforms the other mechanisms considered.
- We propose an approximation technique, based on spanning graphs, that can be used to reduce the number of constraints of the optimization problem and still obtain a geo-indistinguishable mechanism.
- We measure the impact of the approximation on the utility and the number of constraints, and analyze the running time of the whole method, obtaining favorable results.

Plan of the paper The rest of the paper is organized as follows. Next section recall some preliminary notions. In Section 3 we illustrate our method to produce a geo-indistinguishable and optimal mechanism as the solution of a linear optimization problem, and we propose a technique to reduce the number of constraints used in the problem. In Section 4 we evaluate our mechanism with respect to respect to other ones in the literature. Finally, in Section 5, we discuss related work and we conclude.

2 Preliminaries

2.1 Location obfuscation, quality loss and adversary’s error

A common way of achieving location privacy is to apply a *location obfuscation* mechanism, that is a probabilistic function $K : \mathcal{X} \rightarrow \mathcal{P}(\mathcal{X})$ where \mathcal{X} is the set of possible locations, and $\mathcal{P}(\mathcal{X})$ denotes the set of probability distributions over \mathcal{X} . K takes a location x as input, and produces a *reported location* z which is communicated to the service provider. In this paper we generally consider \mathcal{X} to be finite, in which case K can be represented by a stochastic matrix, where k_{xz} is the probability to report z from location x .

A prior distribution $\pi \in \mathcal{P}(\mathcal{X})$ on the set of locations can be viewed either as modelling the behaviour of the user (the *user profile*), or as capturing the adversary’s *side information* about the user. Given a prior π and a metric d on \mathcal{X} , the expected distance between the real and the reported location is:

$$\text{EXPDIST}(K, \pi, d) = \sum_{x,z} \pi_x k_{xz} d(x, z)$$

From the user’s point of view, we want to quantify the *service quality loss* (SQL) produced by the mechanism K . Given a *quality metric* d_Q on locations, such that $d_Q(x, z)$ measures how much the quality decreases by reporting z when

the real location is x (the Euclidean metric d_2 being a typical choice), we can naturally define the quality loss as the expected distance between the real and the reported location, that is $\text{SQL}(K, \pi, d_Q) = \text{EXPDIST}(K, \pi, d_Q)$. The SQL can also be viewed as the (inverse of the) utility of the mechanism.

Similarly, we want to quantify the *privacy* provided by K . A natural approach, introduced in [12] is to consider a Bayesian adversary with some prior information π , trying to remap z back to a guessed location \hat{x} . A remapping strategy can be modelled by a stochastic matrix H , where $h_{z\hat{x}}$ is the probability to map z to \hat{x} . Then the privacy of the mechanism can be defined as the expected error of an adversary under the best possible remapping:

$$\text{ADVErrOR}(K, \pi, d_A) = \min_H \text{EXPDIST}(KH, \pi, d_A)$$

Note that the composition KH of K and H is itself a mechanism. Similarly to d_Q , the metric $d_A(x, \hat{x})$ captures the adversary's loss when he guess \hat{x} when the real location is x . Note that d_Q and d_A can be different, but a natural choice is to use the Euclidean distance for both.

A natural question, then, is to construct a mechanism that achieves *optimal privacy*, given an *SQL constraint*.

Definition 1. *Given a prior π , a quality metric d_Q , a quality bound q and an adversary metric d_A , a mechanism K is q -OPTPRIV(π, d_A, d_Q) iff*

1. $\text{SQL}(K, \pi, d_Q) \leq q$, and
2. for all mechanisms K' , $\text{SQL}(K', \pi, d_Q) \leq q$ implies

$$\text{ADVErrOR}(K', \pi, d_A) \leq \text{ADVErrOR}(K, \pi, d_A)$$

In other words, a q -OPTPRIV mechanism provides the best privacy (expressed in terms of ADVErrOR) among all mechanisms with SQL at most q . This problem was studied in [8], providing a method to construct such a mechanism for any q, π, d_A, d_Q , by solving a properly constructed linear program.

2.2 Differential privacy

Differential privacy was originally introduced in the context of statistical databases, requiring that a query should produce similar results when applied to *adjacent* databases, i.e. those differing by a single row. The notion of adjacency is related to the Hamming metric $d_h(x, x')$ defined as the number of rows in which x, x' differ. Differential privacy requires that the greater the hamming distance between x, x' is, the more distinguishable they are allowed to be.

This concept can be naturally extended to any set of secrets \mathcal{X} , equipped with a metric d_x [13,11]. The distance $d_x(x, x')$ expresses the *distinguishability level* between x and x' : if the distance is small then the secrets should remain indistinguishable, while secrets far away from each other are allowed to be distinguished by the adversary. The metric should be chosen depending on the application at hand and the semantics of the privacy notion that we try to achieve.

Following the notation of [11], a mechanism is a probabilistic function $K : \mathcal{X} \rightarrow \mathcal{P}(\mathcal{Z})$, where \mathcal{Z} is a set of *reported values* (assumed finite for the purposes of this paper). The similarity between probability distributions can be measured by the multiplicative distance $d_{\mathcal{P}}$ defined as $d_{\mathcal{P}}(\mu_1, \mu_2) = \sup_{z \in \mathcal{Z}} |\ln \frac{\mu_1(z)}{\mu_2(z)}|$ with $|\ln \frac{\mu_1(z)}{\mu_2(z)}| = 0$ if both $\mu_1(z), \mu_2(z)$ are zero and ∞ if only one of them is zero. In other words, $d_{\mathcal{P}}(\mu_1, \mu_2)$ is small iff μ_1, μ_2 assign similar probabilities to each value z .

The generalized variant of differential privacy under the metric $d_{\mathcal{X}}$, called $d_{\mathcal{X}}$ -privacy, is defined as follows:

Definition 2. A mechanism $K : \mathcal{X} \rightarrow \mathcal{P}(\mathcal{Z})$ satisfies $d_{\mathcal{X}}$ -privacy iff:

$$d_{\mathcal{P}}(K(x), K(x')) \leq d_{\mathcal{X}}(x, x') \quad \forall x, x' \in \mathcal{X}$$

or equivalently $K(x)(z) \leq e^{d_{\mathcal{X}}(x, x')} K(x')(z)$ for all $x, x' \in \mathcal{X}, z \in \mathcal{Z}$. A privacy parameter ϵ can also be introduced by scaling the metric $d_{\mathcal{X}}$ (note that $\epsilon d_{\mathcal{X}}$ is itself a metric).

Differential privacy can then be expressed as ϵd_h -privacy. Moreover, different metrics give rise to various privacy notions of interest; several examples are given in [11].

2.3 Geo-indistinguishability

In the context of location based systems the secrets \mathcal{X} are locations, and we can obtain a useful notion of location privacy by naturally using the Euclidean distance d_2 , scaled by a security parameter ϵ . The resulting notion of ϵd_2 -privacy, called ϵ -geo-indistinguishability in [9], requires that a location obfuscation mechanism should produce similar results when applied to locations that are geographically close. This prevents the service provider from inferring the user's location with accuracy, while allowing him to get approximate information required to provide the service. [9] studies this notion in detail, arguing that it provides a natural notion of location privacy that is independent from the prior information of the adversary.

Moreover, [9] shows that geo-indistinguishability can be achieved by adding noise to the user's location drawn from a 2-dimensional Laplace distribution. This can be easily done in polar coordinates by selecting an angle uniformly and a radius from a Gamma distribution. If a restricted set of reported locations is allowed, then the location produced by the mechanism can be mapped back to the closest among the allowed ones.

Although the Laplace mechanism provides an easy and practical way of achieving geo-indistinguishability, independently from any user profile, its utility is not always optimal. In the next section we show that by tailoring a mechanism to a prior corresponding to a specific user profile, we can achieve better utility for that prior, while still satisfying geo-indistinguishability, i.e. a privacy guarantee independent from the prior. The evaluation results in Section 4 show that the optimal mechanism can provide substantial improvements compared to the Laplace mechanism.

3 Geo-indistinguishable mechanisms of optimal utility

As discussed in the introduction, our main goal is, given a set of locations \mathcal{X} with a privacy metric $d_{\mathcal{X}}$ (typically the Euclidean distance), a privacy level ϵ , a user profile π and a quality metric d_Q , to find an $\epsilon d_{\mathcal{X}}$ -private mechanism such that its SQL is as small as possible.

We start by describing a set of linear constraints that enforce $\epsilon d_{\mathcal{X}}$ -privacy, which allows to obtain an optimal mechanism as a linear optimization problem. However, the number of constraints can be large, making the approach computationally demanding as the number of locations increases. As a consequence, we propose an approximate solution that replaces $d_{\mathcal{X}}$ with the metric induced by a spanning graph. We discuss a greedy algorithm to calculate the spanning graph and analyze its running time. Finally, we show that, if the quality and adversary metrics coincide, then the constructed (exact or approximate) mechanisms also provide optimal privacy in terms of ADVELOCITY.

3.1 Constructing an optimal mechanism

The constructed mechanism is assumed to have as both input and output a predetermined finite set of locations \mathcal{X} . For instance, \mathcal{X} can be constructed by dividing the map in a finite number of regions (of arbitrary size and shape), and selecting in \mathcal{X} a representative location for each region. We also assume a prior π over \mathcal{X} , representing the probability of the user being at each location at any given time.

Given a privacy metric $d_{\mathcal{X}}$ (typically the Euclidean distance) and a privacy parameter ϵ , the goal is to construct a $\epsilon d_{\mathcal{X}}$ -private mechanism K such that the *service quality loss* with respect to a quality metric d_Q is minimum. This property is formally defined below:

Definition 3. *Given a prior π , a privacy metric $d_{\mathcal{X}}$, a privacy parameter ϵ and a quality metric d_Q , a mechanism K is $\epsilon d_{\mathcal{X}}$ -OPTSQL(π, d_Q) iff:*

1. K is $\epsilon d_{\mathcal{X}}$ -private, and
2. for all mechanisms K' , if K' is $\epsilon d_{\mathcal{X}}$ -private then

$$\text{SQL}(K, \pi, d_Q) \leq \text{SQL}(K', \pi, d_Q)$$

Note that $\epsilon d_{\mathcal{X}}$ -OPTSQL optimizes SQL given a privacy constraint, while q -OPTPRIV (Definition 1) optimizes privacy, given an SQL constraint.

In order for K to be $\epsilon d_{\mathcal{X}}$ -private it should satisfy the following constraints:

$$k_{xz} \leq e^{\epsilon d_{\mathcal{X}}(x, x')} k_{x'z} \quad x, x', z \in \mathcal{X}$$

Hence, we can construct an optimal mechanism by solving a linear optimization problem, minimizing $\text{SQL}(K, \pi, d_Q)$ while satisfying $\epsilon d_{\mathcal{X}}$ -privacy:

$$\begin{aligned}
\textbf{Minimize:} \quad & \sum_{x,z \in \mathcal{X}} \pi_x k_{xz} d_Q(x, z) \\
\textbf{Subject to:} \quad & k_{xz} \leq e^{\epsilon d_{\mathcal{X}}(x, x')} k_{x'z} & x, x', z \in \mathcal{X} \\
& \sum_{z \in \mathcal{X}} k_{xz} = 1 & x \in \mathcal{X} \\
& k_{xz} \geq 0 & x, z \in \mathcal{X}
\end{aligned}$$

It is easy to see that the mechanism K generated by the previous optimization problem is $\epsilon d_{\mathcal{X}}$ -OPTSQL(π, d_Q).

3.2 A more efficient method using spanners

In the optimization problem of the previous section, the $\epsilon d_{\mathcal{X}}$ -privacy definition introduces $|\mathcal{X}|^3$ constraints in the linear program. However, in order to be able to manage a large number of locations, we would like to reduce this amount to a number in the order of $O(|\mathcal{X}|^2)$. One possible way to achieve this is to use the *dual form* of the linear program (shown in the appendix). The dual program has as many constraints as variables in the primal program (in this case $|\mathcal{X}|^2$) and one variable for each constraint in the primal program (in this case $O(|\mathcal{X}|^3)$). Since the primal linear program finds the optimal solution in a finite number of steps, it is guaranteed by the strong duality theorem that dual program will also do so. However, as shown in Section 4.2, in practice the dual program does not offer a substantial improvement with respect to the primal one (a possible explanation being that, although fewer in number, the constraints in the dual program are more complex, in the sense that they each involve a larger number of variables).

An alternative approach is to exploit the structure of the metric $d_{\mathcal{X}}$. So far we are not making any assumption about $d_{\mathcal{X}}$, and therefore we need to specify $|\mathcal{X}|$ constraints for each pair of locations x and x' . However, it is worth noting that if the distance $d_{\mathcal{X}}$ is induced by a weighted graph (i.e. the distance between each pair of locations is the weight of a minimum path in a graph), then we only need to consider $|\mathcal{X}|$ constraints for each pair of locations that are *adjacent in the graph*. An example of this is the usual definition of differential privacy: since the adjacency relation between databases induces the Hamming distance d_h , we only need to require the differential privacy constraint for each pair of databases that are adjacent in the Hamming graph (i.e. that differ in one individual).

It might be the case, though, that the metric $d_{\mathcal{X}}$ is not induced by any graph (other than the complete graph), and consequently the amount of constraints remains the same. In fact, this is generally the case for the Euclidean metric. Therefore, we consider the case in which $d_{\mathcal{X}}$ can be *approximated* by some graph-induced metric.

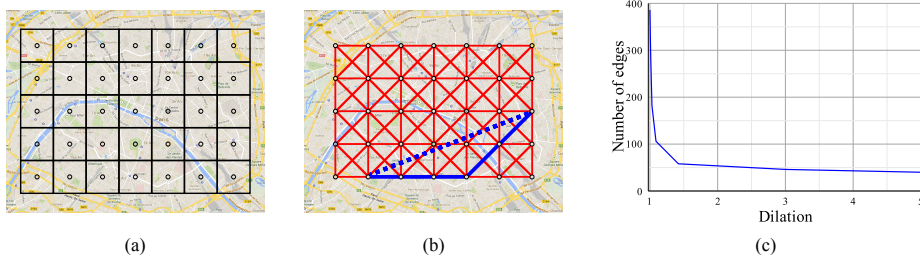


Fig. 1. (a) a division of the map of Paris into a 7×5 square grid. The set of locations \mathcal{X} contains the centers of the regions. (b) A spanner of \mathcal{X} with dilation $\delta = 1.08$. (c) Relation between the number of edges and the dilation for the presented spanner.

If G is an undirected weighted graph, we denote with d_G the distance function induced by G , i.e. $d_G(x, x')$ denotes the weight of a minimum path between the nodes x and x' in G . Then, if the set of nodes of G is \mathcal{X} and the weight of its edges is given by the metric $d_{\mathcal{X}}$, we can approximate $d_{\mathcal{X}}$ with d_G . In this case, we say that G is a spanning graph, or a spanner [14,15], of \mathcal{X} .

Definition 4 (Spanner). A weighted graph $G = (\mathcal{X}, E)$, with $E \subseteq \mathcal{X} \times \mathcal{X}$ and weight function $w : E \rightarrow \mathbb{R}$ is a spanner of \mathcal{X} if

$$w(x, x') = d_{\mathcal{X}}(x, x') \quad \forall (x, x') \in E$$

Note that if G is a spanner of \mathcal{X} , then

$$d_G(x, x') \geq d_{\mathcal{X}}(x, x') \quad \forall x, x' \in \mathcal{X}$$

A main concept in the theory of spanners is that of dilation, also known as stretch factor:

Definition 5 (Dilation). Let $G = (\mathcal{X}, E)$ be a spanner of \mathcal{X} . The dilation of G is calculated as:

$$\delta = \max_{x \neq x' \in \mathcal{X}} \frac{d_G(x, x')}{d_{\mathcal{X}}(x, x')}$$

A spanner of \mathcal{X} with dilation δ is also called a δ -spanner of \mathcal{X} .

Informally, a spanner of \mathcal{X} with dilation δ can be considered as an approximation of the metric $d_{\mathcal{X}}$ in which distances between nodes are “stretched” by a factor of at most δ . Spanners are generally used to approximate distances in a geographic network without considering the individual distances between each pair of nodes. An example of a spanner for a grid in the map can be seen in Figure 1.

If G is a δ -spanner of \mathcal{X} , then we can see that the following property holds:

$$d_G(x, x') \leq \delta d_{\mathcal{X}}(x, x') \quad \forall x, x' \in \mathcal{X}$$

With this property we can state the following proposition:

Proposition 1. *Let \mathcal{X} be a set of locations with metric $d_{\mathcal{X}}$, and let G be a δ -spanner of \mathcal{X} . If a mechanism K for \mathcal{X} is $\frac{\epsilon}{\delta}d_G$ -private, then K is $\epsilon d_{\mathcal{X}}$ -private.*

We can then propose a new optimization problem to obtain a $\epsilon d_{\mathcal{X}}$ -private mechanism. If $G = (\mathcal{X}, E)$ is a δ -spanner of \mathcal{X} , we require not the constraints corresponding to $\epsilon d_{\mathcal{X}}$ -privacy, but those corresponding to $\frac{\epsilon}{\delta}d_G$ -privacy instead, that is, $|\mathcal{X}|$ constraints for each edge of G :

$$\text{Minimize: } \sum_{x, z \in \mathcal{X}} \pi_x k_{xz} d_Q(x, z)$$

$$\begin{aligned} \text{Subject to: } k_{xz} &\leq e^{\frac{\epsilon}{\delta} d_G(x, x')} k_{x'z} & z \in \mathcal{X}, (x, x') \in E \\ \sum_{x \in \mathcal{X}} k_{xz} &= 1 & x \in \mathcal{X} \\ k_{xz} &\geq 0 & x, z \in \mathcal{X} \end{aligned}$$

Since the resulting mechanism is $\frac{\epsilon}{\delta}d_G$ -private, by Proposition 1 it must also be $\epsilon d_{\mathcal{X}}$ -private. However, the number of constraints induced by $\frac{\epsilon}{\delta}d_G$ -privacy is now $|E||\mathcal{X}|$. Moreover, as discussed in the next section, for any $\delta > 1$ there is an algorithm that generates a δ -spanner with $O(\frac{|\mathcal{X}|}{\delta-1})$ edges, which means that, fixing δ , the total number of constraints of the linear program is $O(|\mathcal{X}|^2)$.

It is worth noting that although $\epsilon d_{\mathcal{X}}$ -privacy is guaranteed, optimality is lost: the obtained mechanism is $\frac{\epsilon}{\delta}d_G$ -OPTSQL(π, d_Q) but not necessarily $\epsilon d_{\mathcal{X}}$ -OPTSQL(π, d_Q), since the set of $\frac{\epsilon}{\delta}d_G$ -private mechanisms is a subset of the set of $\epsilon d_{\mathcal{X}}$ -private mechanisms. The SQL of the obtained mechanism will now depend on the dilation δ of the spanner: the smaller δ is, the closer the SQL of the mechanism will be from the optimal one. However, if δ is too small then the number of edges of the spanner will be large, and therefore the number of constraints in the linear program will increase. In fact, when $\delta = 1$ the mechanism obtained is also $\epsilon d_{\mathcal{X}}$ -OPTSQL(π, d_Q) (since d_G and $d_{\mathcal{X}}$ coincide), but the amount of constraints is in general $O(|\mathcal{X}|^3)$. In consequence, there is a tradeoff between the accuracy of the approximation and the number of constraints in linear program.

3.3 An algorithm to construct a δ -spanner

The previous approach requires to compute a spanner for \mathcal{X} . Moreover, given a dilation factor δ , we are interested in generating a δ -spanner with a reasonably small number of edges. In this section we describe a simple greedy algorithm to get a δ -spanner of \mathcal{X} , presented in [14]. This procedure (described in Algorithm 1) is a generalization of Kruskal's minimum spanning tree algorithm.

The idea of the algorithm is the following: we start with a spanner with an empty set of edges (lines 2-3). In the main loop we consider all possible edges

Algorithm 1 Algorithm to get a δ -spanner of \mathcal{X}

```

1: procedure GETSPANNER( $\mathcal{X}, d_{\mathcal{X}}, \delta$ )
2:    $E := \emptyset$ 
3:    $G := (\mathcal{X}, E)$ 
4:   for all  $(x, x') \in (\mathcal{X} \times \mathcal{X})$  do                                 $\triangleright$  taken in increasing order wrt  $d_{\mathcal{X}}$ 
5:     if  $d_G(x, x') > \delta d_{\mathcal{X}}(x, x')$  then
6:        $E := E \cup \{(x, x')\}$ 
7:     end if
8:   end for
9:   return  $G$ 
10: end procedure

```

(that is, all pairs of locations) in *increasing order* with respect to the distance function $d_{\mathcal{X}}$ (lines 4-8), and if the weight of a minimum path between the two corresponding locations in the current graph is bigger than δ times the distance between them, we add the edge to the spanner. By construction, at the end of the procedure, graph G is a δ -spanner of \mathcal{X} .

A crucial result presented in [14] is that, in the case where \mathcal{X} is a set of points in the Euclidean plane, the degree of each node in the generated spanner only depends on the dilation factor:

Theorem 1. *Let $\delta > 1$. If G is a δ -spanner for $\mathcal{X} \subseteq \mathbb{R}^2$, with the Euclidean distance d_2 as metric, then the degree of each node in the spanner constructed by Algorithm 1 is $O(\frac{1}{\delta-1})$.*

This result is useful to estimate the total number of edges in the spanner, since our goal is to generate a *sparse* spanner, i.e. a spanner with $O(|\mathcal{X}|)$ edges.

Considering the running time of the algorithm, since the main loop requires all pair of regions to be sorted increasingly by distance, we need to perform this sorting before the loop. This step takes $O(|\mathcal{X}|^2 \log |\mathcal{X}|)$. The main loop performs a minimum-path calculation in each step, with $|\mathcal{X}|^2$ total steps. If we use, for instance, Dijkstra's algorithm, each of these operations can be done in $O(|E| + |\mathcal{X}| \log |\mathcal{X}|)$. If we select δ so that the final amount of edges in the spanner is linear, i.e. $|E| = O(|\mathcal{X}|)$, we can conclude that the total running time of the main loop is $O(|\mathcal{X}|^3 \log |\mathcal{X}|)$. This turns out to be also the complexity of the whole algorithm.

A common problem in the theory of spanners is the following: given a set of points $\mathcal{X} \subseteq \mathbb{R}^2$ and a maximum amount of edges m , the goal is to find the spanner with *minimum* dilation with at most m edges. This has been proven to be NP-Hard ([16]). In our case, we are interested in the analog of this problem: given a maximum tolerable dilation factor δ , we want to find a δ -spanner with minimum amount of edges. However, we can see that the first problem can be expressed in terms of the second (for instance, with a binary search on the dilation factor), which means that the second problems must be at least NP-Hard as well.

3.4 ADVERROR of the obtained mechanism

As discussed in 2.1, the privacy of a location obfuscation mechanism can be expressed in terms of ADVERROR for an adversary metric d_A . In [8], the problem of optimizing privacy for a given SQL constraint is studied, providing a method to obtain a q -OPTPRIV(π, d_A, d_Q) mechanism for any q, π, d_Q, d_A .

In our case, we optimize SQL for a given privacy constraint, constructing a ϵd_x -OPTSQL(π, d_Q) mechanism. We now show that, if d_Q and d_A coincide, the mechanism generated by any of the two optimization problems of the previous sections is also q -OPTPRIV(π, d_Q, d_Q).

ADVERROR corresponds to the attacker's remapping H that minimizes its expected error with respect to the metric d_A and its prior knowledge π . The composition KH of a mechanism and its remapping is itself a mechanism. The following result shows that remapping does not violate d_x -privacy.

Lemma 1. *Let K be a d_x -private mechanism, and let H be a remapping. Then KH is d_x -private.*

In the case in which the metric d_A used by the adversary and the quality metric d_Q coincide, we can see that no remapping is needed by the adversary: the mechanism K obtained by solving the optimization problem is d_x -OPTSQL(π, d_Q), and then, by Lemma 1, no remapping H can decrease the SQL:

$$\text{SQL}(K, \pi, d_Q) \leq \text{SQL}(KH, \pi, d_Q) \quad \forall H$$

This, in turn, implies that SQL and ADVERROR coincide for K , and therefore K must be q -OPTPRIV(π, d_Q, d_Q).

Theorem 2. *If a mechanism K is d_x -OPTSQL(π, d_Q) then it is also q -OPTPRIV(π, d_Q, d_Q) for $q = \text{SQL}(K, \pi, d_Q)$.*

It is important to note that Theorem 2 holds for any metric d_x . This means that both mechanisms obtained as result of the optimization problems presented in Sections 3.1 and 3.2 are q -OPTPRIV(π, d_Q, d_Q) – since they are ϵd_x -OPTSQL(π, d_Q) and $\frac{\epsilon}{\delta} d_G$ -OPTSQL(π, d_Q) respectively – however for a different value of q . In fact, in contrast to the method of [8] in which the quality bound q is given as a parameter, our method optimizes the SQL given a privacy bound. Hence, the resulting mechanism will be q -OPTPRIV(π, d_Q, d_Q), but for a q that is not known in advance (and will depend on the privacy constraint ϵ and the dilation factor δ : the higher the value of ϵ (i.e. the higher the privacy), the lower q will be. Similarly, for a fixed ϵ , the lower the value of δ (i.e. the better the approximation), the lower the SQL of K).

Finally, we must remark that this result only holds in the case where the metrics d_Q, d_A coincide. If the metrics differ, e.g. the quality is measured in terms of the Euclidean distance (the user is interested in accuracy) but the adversary uses the binary distance (he is only interested in the exact location), then this property will no longer be true.

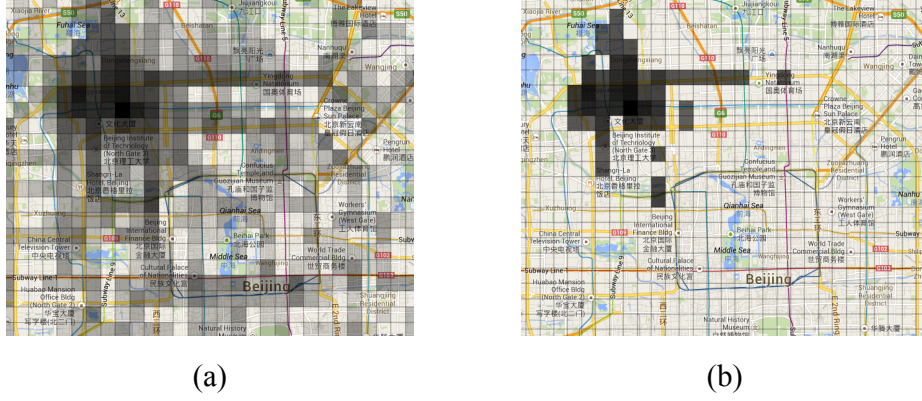


Fig. 2. (a) Division of the map of Beijing into regions of size 0.658 x 0.712 km. The density of each region represents how often individuals from the dataset visit it. (b) The 50 selected regions. These regions are the ones with highest density between the whole set of regions.

4 Evaluation

Given a set of locations \mathcal{X} in the map with a corresponding metric $d_{\mathcal{X}}$, a dilation factor δ , a user u with user profile π and a privacy constraint ϵ , we are able to get a ϵ/δ -OPTSQL(π, d_Q) mechanism for a given quality metric d_Q . In this section we evaluate the location privacy provided by our mechanisms, and compare them with the mechanism of Shorki et al. We consider the construction of the mechanisms under different user profiles, and we compare the privacy offered by them under different prior distributions.

In order to perform a realistic analysis, we construct the different user profiles and prior distributions using the information of several traces from real users, collected in the GeoLife GPS Trajectories dataset ([17], [18], [19]). This dataset contains 17621 traces from 182 users, moving mainly in the north-west of Beijing, China, in a period of over five years (from April 2007 to August 2012). The traces show users performing routinary tasks (like going to and from work), and also traveling, shopping, and doing other kinds of entertainment or unusual activities. Besides, the traces were logged by users using different means of transportation, like walking, public transport or bike. More than 90% of the traces were logged in a dense representation, which means that the individual points in the trace were reported every 1-5 seconds or every 5-10 meters.

For the purposes of evaluation, we divide the map of Beijing into a grid of regions 0.6583 km wide and 0.7116 km high (Figure 2a). We measure the “popularity” of each region with respect to all the individuals of the dataset as follows:

- For each user, we calculate the number of points in the traces of this user that falls in each region. We refer to this number as the *rank* of each region.

We then select the 30 regions with highest ranks for each user (without considering regions with a rank of 0).

- For each region, we assign a score: the number of users that have this regions in their highest ranked ones. The 50 selected regions are those with the highest scores.

Figure 2a shows the division of the map into regions, with the opacity representing the score of each of them, while Figure 2b shows the 50 regions with highest score. We can see that most of the selected regions are located in the south-east of the Haidian district, and all of them are located in the north-west of Beijing. We consider the set of locations \mathcal{X} to be the centers of the selected regions, and the metric $d_{\mathcal{X}}$ to be the Euclidean distance between these centers, i.e. $d_{\mathcal{X}} = d_2$.

4.1 Comparing privacy

Given a given user u with profile π , we are interested in comparing the privacy guarantees offered by the different mechanism presented in previous sections, under different kinds of prior information that might be available to the attacker. Since we need these profiles and prior distributions to reflect accurately the distribution of the users over the map, we only consider users with more than 100 traces logged in the dataset (23% of the individuals in the dataset meet this criteria) within a time period of more than one month but less than one year (40% of the total number of users). From these traces, and in order to only take into account the “recent” behaviour of the user (like in the experiments performed in [8]), we just consider those corresponding to the last month in which the user logged some activity.

The evaluation process is performed as follows: we first select randomly a user u meeting the criteria described before and generate the user profile π of this user. This is done as follows:

- For each region, we consider how many traces include a point inside that region (if a trace has several points in the same region, we consider only one of them).
- We normalize the values and get a probability distribution.

Then, in order to perform a fair comparison, we construct the mechanisms in such a way that their SQL coincide. The first step is to select a privacy level ϵ and a dilation δ , and then construct the mechanism described in Section 3.2. We will call this mechanism OPTSQL. We then set $q = \text{SQL}(\text{OPTSQL}, \pi, d_2)$ and construct the mechanism described in [8], fixing the SQL as q . We call this mechanism OPTPRIV. Finally, we compute a discretized version of the Planar Laplacian mechanism of Andrés et al [9]. under a privacy constraint ϵ' , where ϵ' is selected such that the SQL of this mechanism is also q . We call this mechanism PL,

Our goal is to compare the location privacy offered by these three mechanisms. We note, however, that in general location privacy mechanisms do not

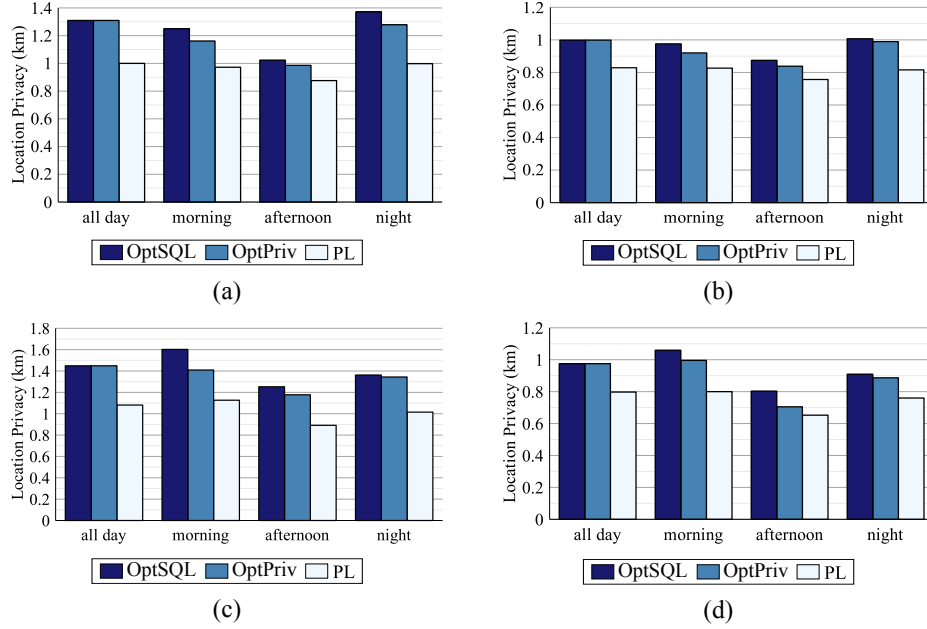


Fig. 3. Privacy of the mechanisms for two different users and under four different priors. (a) Comparison for u_1 , with $\epsilon = 0.5$, and SQL set at 1.31 km. (b) Comparison for u_1 , with $\epsilon = 1.07$, and SQL set at 1 km. (c) Comparison for u_2 , with $\epsilon = 0.5$, and SQL set at 1.45 km. (d) Comparison for u_2 , with $\epsilon = 1.07$, and SQL set at 0.98 km.

satisfy $\epsilon d_{\mathcal{X}}$ -privacy unless they are specifically designed to do so. Therefore, we measure the privacy using the metric ADVERROR, proposed in [8] and described in Section 2.1, which measures the expected error of the attacker under a given prior distribution. Besides, we will compare the privacy offered by these mechanisms under different prior distributions. These distributions represent the information the attacker might have about the likelihood of the user being in each of the considered regions. This prior knowledge might come not only from previous traces of the user, but also from different kinds of side-information accessible to the attacker. For instance, if the movement patterns of the user in the morning are different from those in the night, then the attacker can improve its prior distribution by only considering those traces logged in the morning. In this evaluation, we consider four different prior distributions: the one that comes from all the traces of the user in the considered month (this prior distribution coincides with the user profile), and the ones derived from the traces in the morning (from 7am to noon), afternoon (from noon to 7pm) and night (from 7pm to 7am).

Figure 3 shows the location privacy (in kilometers) offered by the three different mechanism under the four prior distribution mentioned before, for two randomly selected individuals in the dataset, that we will refer as u_1 and u_2 . For each user we perform two experiments, differing in the privacy constraint

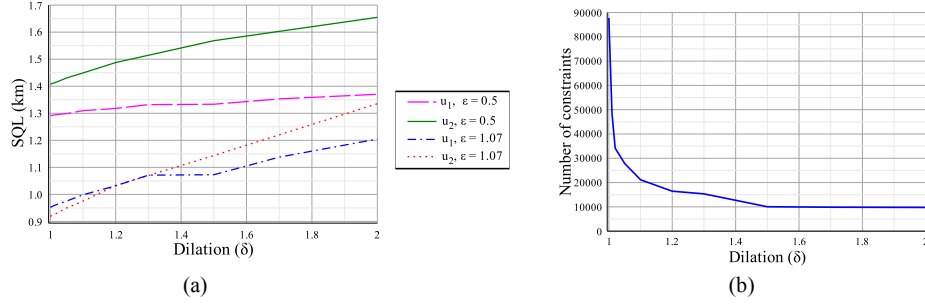


Fig. 4. (a) Relation between SQL and dilation for the four variants of the mechanism OPTSQL with privacy constraints $\epsilon = 0.5$ and $\epsilon = 1.07$, and constructed under the user profiles of u_1 and u_2 . (b) Relation between number of constraints in the optimization problem and dilation of the spanner for the set of considered locations. The spanner is calculated with the greedy algorithm presented in Section 3.3.

used to construct the mechanism OPTSQL: in the first case, we set $\epsilon = 0.5$ (Figures 3a and 3c), while in the second we set $\epsilon = 1.07$ (Figures 3b and 3d). We can see that in all four cases and for all priors considered, the location privacy offered by OPTSQL is higher than that of the other mechanisms, with the only exception being the all-day prior (which is the one used in the construction of the mechanisms) since, as explained in Section 3.4, OPTSQL and OPTPRIV are q -OPTPRIV(π, d_2, d_2) and therefore offer the same privacy.

It can also be observed that in some cases, like in Figure 3a, the location privacy for a more specific prior distribution, like the morning prior, can be higher than the privacy for a more general one, in this case the prior for all day. However this has a simple explanation: since the prior used to construct the mechanism is different than the one being considered, then the mechanism might become less accurate (i.e. it might have a higher SQL) for the more specific prior, and as a consequence it could increase the expected error of the attacker.

4.2 Performance of the approximation algorithm

We recall from Section 3.2 that if we consider a large number of locations in \mathcal{X} , then the number of constraints in the linear program might be high. Hence, we introduced a method based on a spanning graph G to reduce the total number of constraints of the linear program. However, the obtained mechanism might no longer be $\epsilon d_{\mathcal{X}}$ -OPTSQL(π, d_Q), and in fact it has in general a higher SQL than the optimal one.

In this section study how this approximation affects the utility of our mechanism. We consider the construction of the mechanism OPTSQL for users u_1 and u_2 , under privacy constraints $\epsilon = 0.5$ and $\epsilon = 1.07$, and measure the SQL for different values of the dilation δ of the spanner, ranging from 1 (which means that $d_{\mathcal{X}}$ and d_G are the same) to 2. The results can be observed in Figure 4a. We can see that the increase on the SQL in each case is steady, but the speed of the

Table 1. Execution times of our approach for 50 and 75 locations, for different values of δ , and using different methods to solve the linear program.

| | | Primal simplex | | Dual simplex | | Interior-point | | OPTPRIV |
|---------------|----------------|----------------|---------|--------------|---------|----------------|---------|---------|
| | | Primal LP | Dual LP | Primal LP | Dual LP | Primal LP | Dual LP | |
| 50 regions | $\delta = 1.0$ | 57s | X | 40s | 45s | 49m 20s | NI | 1m |
| | $\delta = 1.1$ | 46.4s | 5.2 | 5.9s | 15.5s | 7.5s | NI | |
| | $\delta = 1.2$ | 4m 37s | 2s | 4s | X | 2.7s | NI | |
| | $\delta = 1.5$ | 2s | 1s | 2s | 3s | 0.5s | NI | |
| | $\delta = 2.0$ | Error | 1s | 2s | 2s | 0.5s | NI | |
| 75 regions | $\delta = 1.0$ | X | X | 29m 26s | X | X | NI | 11m |
| | $\delta = 1.1$ | X | Error | 1m 12s | 2m 19s | 55s | NI | |
| | $\delta = 1.2$ | X | Error | 42s | 48.4s | 11.7s | NI | |
| | $\delta = 1.5$ | X | 5m 55s | 19.2s | X | 2.2s | NI | |
| | $\delta = 2.0$ | X | 21.8s | 27.2s | 15.5s | 1.7s | NI | |

increase varies from case to case. Recall that in the experiments performed in Section 4.1 the dilation factor was set at $\delta = 1.1$. We can see that the increase in the SQL goes from 18.6 meters in the best case (u_1 with $\epsilon = 0.5$) to 56 meters in the worst (u_2 with $\epsilon = 1.07$), which represent increases of 1.45% and 6.1% respectively. Therefore, we can conclude that a reasonably small dilation does not produce an important decrease in the utility.

Now, since the goal of using an approximate distance was to reduce the total number of constraints in the optimization problem, we study the relation between this number and the dilation of the spanner. Note that the amount of constraints is independent from the user profile and the privacy level ϵ , and therefore it is enough to consider only one of the variants of OPTSQL presented before. From the results, shown in Figure 4b, it can be seen that the amount of constraints decreases sharply from $\delta = 1$ to $\delta = 1.1$, and more gradually after that. For this particular set of locations \mathcal{X} , the optimization problem for $\delta = 1$ needs to consider 87750 constraints, while with $\delta = 1.1$ this amount decreases to 21150. This represents a reduction of 76% in the number of constraints of the linear program. It is safe to say then that a dilation of 1.1 allows an important reduction of the number of constraints without affecting to much the SQL of the generated mechanism.

Finally, we measure the running time of the method used to generate OPTSQL, under different approaches to solve the linear optimization problem. The experiments were performed in a 2.8 GHz Intel Core i7 MacBook Pro with 8 GB of RAM running Mac OS X 10.9.1, and the source code for the method was written in C++, using the routines in the GLPK library for the linear program. We compare the performance of three different methods included in the library: the simplex method in both its primal and dual form, and the primal-dual interior-point method. Besides, we run these methods on both the primal linear program presented in Section 3.2 and its dual form, presented in Appendix B. Since the running time depends mainly on the number of locations being considered, in the experiments we focus on the case for user u_1 and privacy level $\epsilon = 1.07$. The

results can be seen in Table 1. Some fields are marked with X, meaning that the execution took more than one hour, after which it was stopped. Others are marked with “Error”, meaning that the execution stopped before one hour with an error⁵. A particular case of error happened when running the interior-point method on the dual linear program, where all executions ended with a “numerical instability” error. From the results we can observe that:

- The only two methods that behave consistently (that never finish with error, and the running time increases when the dilation decreases) are the dual simplex and the interior-point methods, both when applied to the primal program.
- From these, the interior-point method performs better in the case of bigger dilation, while it does it much worse for very small ones.
- Somewhat surprisingly, the dual linear program does not offer a significant performance improvement, specially when compared with the interior-point method.

In the case of OPTPRIV, the mechanism is generated using Matlab’s linear program solver (source code kindly provided by the authors of [8]). We can observe that, for dilations of 1.1 and up, our method offers an improvement in terms of running time.

5 Conclusion and related work

Related work. In the last years, a large number of location-privacy protection techniques, diverse both in nature and goals, have been proposed and studied. Many of these aim at allowing the user of an LBS to hide his *identity* from the service provider. Several approaches are based in the notion of k -anonymity [20,21,22], requiring that the attacker cannot identify a user from at least other $k - 1$ different users. Others are based on the idea of letting the users use pseudonyms to interact with the system, and on having regions (*mix zones*, [4,6]), where the users can change their pseudonyms without being traced by the system. All these approaches are incomparable with ours, since ours aims at hiding the *location* of the user and not his identity.

Many approaches to location privacy are based on obfuscating the position of the user. A common technique for this purpose is *cloaking* [23,24,25,21], which consists in blurring the user’s location by reporting a region to the service provider. Another technique is based on adding *dummy locations* [26,27,5] to the request sent to the service provider. In order to preserve privacy, these dummy location should be generated in such a way that they look equally likely to be the user’s real position. Collaborative models were also proposed, where privacy is achieved with a peer-to-peer scheme where users avoid querying the service provider whenever they can find the requested information among their

⁵ The actual error message in this case was: “Error: unable to factorize the basis matrix (1). Sorry, basis recovery procedure not implemented yet”

peers [28]. Finally, in [29] a technique to generate optimal mechanisms under bandwidth and quality constraints is presented. The obtained mechanisms can be based either on dummy locations, cloaking or simple obfuscation.

Differential Privacy has also been used in the context of location privacy. However, it is in general used to protect *aggregate* location information. For instance, [30] presents a way to statistically simulate the location data from a database while providing privacy guarantees. In [31], a quadtree spatial decomposition technique is used to achieve differential privacy in a database with location pattern mining capabilities. Still, there are works that propose the use of differential privacy for the purposes of hiding just an individual’s location information. Dewri [32] proposes a combination of differential privacy and k -anonymity which requires that the distances between the probability distributions corresponding to k fixed locations (defined as the anonymity set) should not be greater than the privacy parameter ϵ .

In [33] the authors use the same generalized notion of differential privacy used in this paper in order to construct a *fair* mechanism that produces similar reported values for “similar” users. Here, the similarity between users is captured by the metric, which is the one used in the generalization. As in this paper, the mechanism is obtained by solving an optimization problem. However, no technique is used to reduce the number of constraints of the linear program.

Conclusion In this paper we have developed a method to generate a mechanism for location privacy that combines the advantages of the geo-indistinguishability privacy guarantee of [9] and the optimal mechanism of [8]. Since linear optimization is computationally demanding, we have provided a technique to reduce the total number of constraints in the linear program, based on the use of a spanning graph to approximate distances between locations, which allows a huge reduction on the number of constraints with only a small decrease in the utility. Finally, we have evaluated the proposed approach using traces from real users, and we have compared both the privacy and the running time of our mechanism with that of [8]. It turns out that our mechanism offers better privacy guarantees when the side knowledge of the attacker is different from the distribution used to construct the mechanisms. Besides, for a reasonably good approximation factor, we have showed that our approach performs much better in terms of running time.

References

1. Freudiger, J., Shokri, R., Hubaux, J.P.: Evaluating the privacy risk of location-based services. In: Proc. of FC’11. Volume 7035 of LNCS., Springer (2011) 31–46
2. Golle, P., Partridge, K.: On the anonymity of home/work location pairs. In: Proc. of PerCom’09. Volume 5538 of LNCS (LNCS). Springer-Verlag (2009) 390–397
3. Krumm, J.: Inference attacks on location tracks. In: Proceedings of the 5th Int. Conf. on Pervasive Computing. Volume 4480 of LNCS., Springer (2007) 127–143
4. Beresford, A.R., Stajano, F.: Location privacy in pervasive computing. IEEE Pervasive Computing **2**(1) (2003) 46–55

5. Chow, R., Golle, P.: Faking contextual data for fun, profit, and privacy. In: Proc. of PES, ACM (2009) 105–108
6. Freudiger, J., Shokri, R., Hubaux, J.P.: On the optimal placement of mix zones. In: Proc. of PETS 2009. Volume 5672 of LNCS., Springer (2009) 216–234
7. Hoh, B., Gruteser, M., Xiong, H., Alrabady, A.: Preserving privacy in gps traces via uncertainty-aware path cloaking. In: Proc. of CCS, ACM (2007) 161–171
8. Shokri, R., Theodorakopoulos, G., Troncoso, C., Hubaux, J.P., Boudec, J.Y.L.: Protecting location privacy: optimal strategy against localization attacks. In: Proc. of CCS, ACM (2012) 617–627
9. Andrés, M.E., Bordenabe, N.E., Chatzikokolakis, K., Palamidessi, C.: Geo-indistinguishability: differential privacy for location-based systems. In: Proc. of CCS, ACM (2013) 901–914
10. Dwork, C., Mcsherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Proc. of TCC. Volume 3876 of LNCS., Springer (2006) 265–284
11. Chatzikokolakis, K., Andrés, M.E., Bordenabe, N.E., Palamidessi, C.: Broadening the scope of Differential Privacy using metrics. In: Proc. of PETS. Volume 7981 of LNCS., Springer (2013) 82–102
12. Shokri, R., Theodorakopoulos, G., Boudec, J.Y.L., Hubaux, J.P.: Quantifying location privacy. In: Proc. of S&P, IEEE (2011) 247–262
13. Reed, J., Pierce, B.C.: Distance makes the types grow stronger: a calculus for differential privacy. In: Proc. of ICFP, ACM (2010) 157–168
14. Narasimhan, G., Smid, M.: Geometric spanner networks. CUP (2007)
15. : Handbook of Computational Geometry. Elsevier (1999)
16. Klein, R., Kutz, M.: Computing Geometric Minimum-Dilation Graphs is NP-Hard. In: Proc. of GD 2006. Volume 4372., Springer (2006) 196–207
17. Zheng, Y., Li, Q., Chen, Y., Xie, X., Ma, W.Y.: Understanding Mobility Based on GPS Data. In: Proc. of UbiComp 2008. (2008)
18. Zheng, Y., Zhang, L., Xie, X., Ma, W.Y.: Mining interesting locations and travel sequences from GPS trajectories. In: Proc. of WWW 2009. (2009)
19. Zheng, Y., Xie, X., Ma, W.Y.: Geolife: A collaborative social networking service among user, location and trajectory. IEEE Data Eng. Bull. **33**(2) (2010) 32–39
20. Gruteser, M., Grunwald, D.: Anonymous usage of location-based services through spatial and temporal cloaking. In: Proc. of MobiSys, USENIX (2003)
21. Gedik, B., Liu, L.: Location privacy in mobile systems: A personalized anonymization model. In: Proc. of ICDCS, IEEE (2005) 620–629
22. Mokbel, M.F., Chow, C.Y., Aref, W.G.: The new casper: Query processing for location services without compromising privacy. In: Proc. of VLDB, ACM (2006) 763–774
23. Bamba, B., Liu, L., Pesti, P., Wang, T.: Supporting anonymous location queries in mobile environments with privacygrid. In: Proc. of WWW, ACM (2008) 237–246
24. Duckham, M., Kulik, L.: A formal model of obfuscation and negotiation for location privacy. In: Proc. of PERVASIVE. Volume 3468 of LNCS., Springer (2005) 152–170
25. Xue, M., Kalnis, P., Pung, H.: Location diversity: Enhanced privacy protection in location based services. In: Proc. of LoCA. Volume 5561 of LNCS., Springer (2009) 70–87
26. Kido, H., Yanagisawa, Y., Satoh, T.: Protection of location privacy using dummies for location-based services. In: Proc. of ICDE Workshops. (2005) 1248
27. Shankar, P., Ganapathy, V., Iftode, L.: Privately querying location-based services with SybilQuery. In: Proc. of UbiComp, ACM (2009) 31–40

28. Shokri, R., Theodorakopoulos, G., Papadimitratos, P., Kazemi, E., Hubaux, J.P.: Hiding in the mobile crowd: Location privacy through collaboration. In: Proc. of the TDSC, IEEE (2014)
29. Herrmann, M., Troncoso, C., Diaz, C., Preneel, B.: Optimal sporadic location privacy preserving systems in presence of bandwidth constraints. In: Proc. of PES. (2013)
30. Machanavajjhala, A., Kifer, D., Abowd, J.M., Gehrke, J., Vilhuber, L.: Privacy: Theory meets practice on the map. In: Proc. of ICDE, IEEE (2008) 277–286
31. Ho, S.S., Ruan, S.: Differential privacy for location pattern mining. In: Proc. of SPRINGL, ACM (2011) 17–24
32. Dewri, R.: Local differential perturbations: Location privacy under approximate knowledge attackers. IEEE Trans. on Mobile Computing **99**(PrePrints) (2012) 1
33. Dwork, C., Hardt, M., Pitassi, T., Reingold, O., Zemel, R.S.: Fairness through awareness. In: Proc. of ITCS, ACM (2012) 214–226

A Proofs

Proposition 1. *Let \mathcal{X} be a set of locations with metric $d_{\mathcal{X}}$, and let G be a δ -spanner of \mathcal{X} . If a mechanism K for \mathcal{X} is $\frac{\epsilon}{\delta}d_G$ -private, then K is $\epsilon d_{\mathcal{X}}$ -private.*

Proof. This proposition is a direct consequence of the property

$$d_G(x, x') \leq \delta d_{\mathcal{X}}(x, x') \quad \forall x, x' \in \mathcal{X}$$

and one of the results presented in [11], which states that if two metrics $d_{\mathcal{X}}$ and $d_{\mathcal{Y}}$ are such that $d_{\mathcal{X}} \leq d_{\mathcal{Y}}$ (point-wise), then $d_{\mathcal{X}}$ -privacy implies $d_{\mathcal{Y}}$ -privacy. \square

Lemma 1. *Let K be a $d_{\mathcal{X}}$ -private mechanism, and let H be a remapping. Then KH is $d_{\mathcal{X}}$ -private.*

Proof. We know that

$$(KH)_{x\hat{x}} = \sum_{z \in \mathcal{X}} k_{xz} h_{z\hat{x}}, \quad \forall x, \hat{x} \in \mathcal{X}$$

Since K is $\epsilon d_{\mathcal{X}}$ -private, we also know that

$$k_{xz} \leq e^{\epsilon d_{\mathcal{X}}(x, x')} k_{x'z}, \quad \forall x, x', z \in \mathcal{X}$$

Therefore, given $x, x' \in \mathcal{X}$, it holds that for all $\hat{x} \in \mathcal{X}$:

$$\begin{aligned} (KH)_{x\hat{x}} &= \sum_{z \in \mathcal{X}} k_{xz} h_{z\hat{x}} \\ &\leq \sum_{z \in \mathcal{X}} e^{\epsilon d_{\mathcal{X}}(x, x')} k_{x'z} h_{z\hat{x}} \\ &= e^{\epsilon d_{\mathcal{X}}(x, x')} \sum_{z \in \mathcal{X}} k_{x'z} h_{z\hat{x}} \\ &= e^{\epsilon d_{\mathcal{X}}(x, x')} (KH)_{x'\hat{x}} \end{aligned}$$

and therefore KH is $\epsilon d_{\mathcal{X}}$ -private. \square

Theorem 2. *If a mechanism K is $d_{\mathcal{X}}$ -OPTSQL(π, d_Q) then it is also q -OPTPRIV(π, d_Q, d_Q) for $q = \text{SQL}(K, \pi, d_Q)$.*

Proof. Let $d_A = d_Q$. We recall from Section 2.1 that for an arbitrary mechanism M , it holds that

$$\begin{aligned} \text{ADVErrOR}(M, \pi, d_Q) &= \min_H \text{EXPDIST}(MH, \pi, d_Q) \\ &= \min_H \text{SQL}(MH, \pi, d_Q) \end{aligned}$$

which means that

$$\text{ADVErrOR}(M, \pi, d_Q) \leq \text{SQL}(M, \pi, d_Q) \quad (1)$$

Let K be a $d_{\mathcal{X}}$ -OPTSQL(π, d_Q) mechanism. Suppose that

$$\text{ADVErrOR}(K, \pi, d_Q) < \text{SQL}(K, \pi, d_Q)$$

This means that there is a remapping H , other than the identity, such that

$$\text{SQL}(KH, \pi, d_Q) < \text{SQL}(K, \pi, d_Q)$$

However, by Lemma 1 we know that KH is also $d_{\mathcal{X}}$ -private, and therefore, recalling Definition 3, KH would not be $d_{\mathcal{X}}$ -OPTSQL(π, d_Q), which is a contradiction. Therefore, we can state that

$$\text{ADVErrOR}(K, \pi, d_Q) = \text{SQL}(K, \pi, d_Q) \quad (2)$$

Now, in order to see that K is also q -OPTPRIV(π, d_Q, d_Q), with $q = \text{SQL}(K, \pi, d_Q)$, let K' be such that

$$\text{SQL}(K', \pi, d_Q) \leq \text{SQL}(K, \pi, d_Q) \quad (3)$$

According to Definition 1 we need to prove that

$$\text{ADVErrOR}(K', \pi, d_Q) \leq \text{ADVErrOR}(K, \pi, d_Q)$$

And in fact we can see that

$$\begin{aligned} \text{ADVErrOR}(K', \pi, d_Q) &\leq \text{SQL}(K', \pi, d_Q) && \text{(by (1))} \\ &\leq \text{SQL}(K, \pi, d_Q) && \text{(by (3))} \\ &= \text{ADVErrOR}(K, \pi, d_Q) && \text{(by (2))} \end{aligned}$$

which concludes our proof. \square

B Dual form of the optimization problem

In this section we will show dual form of the optimization problem presented in Section 3.2. In order to obtain the dual form we need to consider one variable for each of the constraints in the original linear program that are not constraints on single variables. We recall that the original linear program is as follows:

$$\text{Minimize: } \sum_{x,z \in \mathcal{X}} \pi(x) k_{xz} d_Q(x, z)$$

$$\text{Subject to: } k_{xz} \leq e^{\frac{\epsilon}{\delta} d_G(x, x')} k_{x'z} \quad z \in \mathcal{X}, (x, x') \in E \quad (1)$$

$$\sum_{x \in \mathcal{X}} k_{xz} = 1 \quad x \in \mathcal{X} \quad (2)$$

$$k_{xz} \geq 0 \quad x, z \in \mathcal{X}$$

Therefore, for the dual program we will consider two sets of variables:

- The variables of the form $a_{xx'z}$, with $z \in \mathcal{X}, (x, x') \in E$, corresponding to the constraints in (1).
- The variables of the form b_x , with $x \in \mathcal{X}$, corresponding to the constraints in (2).

The dual linear program is then as follows:

$$\text{Maximize: } \sum_{x \in \mathcal{X}} b_x$$

$$\text{Subject to: } b_x + \sum_{(x, x') \in E} (e^{\frac{\epsilon}{\delta} d_G(x, x')} a_{x'xz} - a_{xx'z}) \leq \pi(x) d_Q(x, z) \quad x, z \in \mathcal{X}$$

$$a_{xx'z} \geq 0 \quad z \in \mathcal{X}, (x, x') \in E$$